Continuous Integration

Group 18

Team B

Olivia Betts
Zac Bhumgara
Nursyarmila Ahmad Shukri
Cameron Duncan-Johal
Muaz Waqas
Oliver Northwood
Teddy Seddon

Our approaches:

a. https://resources.github.com/ci-cd/#:~:text=Continuous%20integration%20(CI)%20automatically%20builds,code%20changes%20to%20customers%20directly.
b. https://github.blog/2022-02-02-build-ci-cd-pipeline-github-actions-four-steps/
c. Runs tests automatically after pushing code to version control.
d. Robust such that any code pushed to the version control will not break the CI.
e. Automatically builds java files (.jar file) after code is pushed to version control.

Continuous integration, CI, is a software development practice where members of a team integrate their work frequently, usually daily, with each integration being verified by an automated build to detect integration errors as quickly as possible. This allows for issues to be detected much sooner and means they are smaller thus easier to solve.

The first step in CI is to ensure that the code runs and shows errors as we planned. It also has to be robust such that any code pushed to the version control will not break the CI. We used Github Actions as our CI server since everyone in our team is already familiar with Github and it is also widely used. A unit-testing code is created so that our code could be tested regularly after pushing code to version control and any errors could be detected immediately. The first workflow is created

Automatically builds java files (.jar file) after code is pushed to version control.
CI begins in shared repositories, where teams collaborate on code using version control systems like Git. A version control system keeps track of code changes and makes them easy to revert if something breaks. It also enables configuration as code, which allows teams to manage testing, infrastructure, and more as versioned artefacts.

One benefit of CI is development velocity. Ongoing feedback enables developers to make minor changes more frequently, instead of waiting for one release. Another is in stability and reliability since automated, continued testing ensures that codebases stay stable and are release ready at any time.

Firstly, we have added an automated build to help developers commit code changes more frequently by running automated tests on every commit.

Secondly, we have also set up a JaCoCo test coverage tool.

Next, we have added a JAR upload to the workflow.

Lastly, Checkstyle has been added to ensure all developers follow a consistent coding standard so that the number of bugs can be reduced.